Tutorial: Modelagem de Objetos e Mensagens a Partir de um Caso de Uso

1 Objetivo

Este tutorial orienta o processo de análise de um caso de uso para extrair objetos, mensagens e suas direções para modelagem orientada a objetos. Ao transformar uma descrição funcional em uma perspectiva orientada a objetos, você pode criar uma base para diagramas de sequência e refinar diagramas de classes.

2 Caso de Uso Exemplo

Considere a seguinte descrição de caso de uso para um sistema de livraria online:

Caso de Uso: Comprar um Livro
"O cliente seleciona um livro do catálogo. O sistema exibe os detalhes do livro. O cliente adiciona o livro ao carrinho. O sistema atualiza o carrinho com o livro selecionado."

3 Análise Passo a Passo

3.1 Passo 1: Identificar Objetos (Instâncias de Classes)

Para identificar objetos, focamos nos substantivos e agentes na descrição do caso de uso. Esses representam entidades que participam do cenário e são candidatos a classes no modelo de domínio.

- **Substantivos no caso de uso**: cliente, livro, catálogo, sistema, carrinho, detalhes do livro.
- **Análise**:
  - **Cliente**: Um ator que interage com o sistema, provavelmente representado como um objeto.
  - **Livro**: Um produto no sistema, um forte candidato a objeto.
  - **Catálogo**: Uma coleção ou interface que contém livros, provavelmente um objeto.
  - **Sistema**: Frequentemente representa a aplicação em si, mas na modelagem

orientada a objetos, focamos em componentes específicos (por exemplo, catálogo ou carrinho) em vez do sistema como um todo.

  - **Carrinho**: Um contêiner para livros selecionados, outro objeto.
  - **Detalhes do livro**: Provavelmente atributos ou uma visão do livro, não um objeto independente.

- **Objetos Identificados**: Cliente, Livro, Catálogo, Carrinho.

3.2 Passo 2: Identificar Mensagens (Métodos)

Em seguida, examinamos os verbos no caso de uso para identificar ações que se traduzem em mensagens (chamadas de métodos) entre objetos.

- **Verbos no caso de uso**: seleciona, exibe, adiciona, atualiza.
- **Análise**:
  - **Seleciona**: O cliente escolhe um livro do catálogo, sugerindo um método como `selecionarLivro()`.
  - **Exibe**: O sistema mostra os detalhes do livro, o que pode ser um método como `exibirDetalhes()`.
  - **Adiciona**: O cliente adiciona o livro ao carrinho, indicando um método como `adicionarLivro()`.
  - **Atualiza**: O carrinho atualiza seu estado, sugerindo um método como `atualizar()`.
- **Mensagens Identificadas**:
  - `selecionarLivro()` (Catálogo)
  - `exibirDetalhes()` (Livro ou Catálogo)
  - `adicionarLivro()` (Carrinho)
  - `atualizar()` (Carrinho)

3.3 Passo 3: Determinar a Direção das Mensagens

Para cada ação, identificamos o emissor (quem inicia a ação) e o receptor (quem executa ou responde à ação). Isso auxilia na construção de diagramas de sequência e na definição de responsabilidades das classes.

- **Ação: "O cliente seleciona um livro do catálogo"**
  - **Emissor**: Cliente
  - **Receptor**: Catálogo
  - **Mensagem**: `selecionarLivro(livroId)`
  - **Explicação**: O cliente inicia a ação ao selecionar um livro, então o objeto Cliente envia uma mensagem ao objeto Catálogo para recuperar ou marcar o livro selecionado.
- **Ação: "O sistema exibe os detalhes do livro"**

- **Emissor**: Catálogo (ou Livro, dependendo do design)
- **Receptor**: Cliente
- **Mensagem**: `exibirDetalhes()`
- **Explicação**: Após a seleção, o Catálogo ou Livro fornece os detalhes do livro ao Cliente.

- **Ação: "O cliente adiciona o livro ao carrinho"**
  - **Emissor**: Cliente
  - **Receptor**: Carrinho
  - **Mensagem**: `adicionarLivro(livro)`
  - **Explicação**: O cliente adiciona o livro ativamente, então o objeto Cliente envia uma mensagem ao objeto Carrinho para incluir o livro.
- **Ação: "O sistema atualiza o carrinho com o livro selecionado"**
  - **Emissor**: Carrinho (ação interna ou desencadeada por `adicionarLivro()`)
  - **Receptor**: Carrinho (auto-mensagem ou atualização interna)
  - **Mensagem**: `atualizar()`
  - **Explicação**: O carrinho atualiza seu estado interno após receber o livro, provavelmente como uma operação interna após a mensagem `adicionarLivro()`.

3.4 Passo 4: Resultados da Análise

Esta análise fornece a base para a modelagem orientada a objetos:

- **Objetos e Classes**: Identificamos Cliente, Livro, Catálogo e Carrinho como objetos principais, sugerindo classes no modelo de domínio.
- **Métodos**: As mensagens (`selecionarLivro()`, `exibirDetalhes()`, `adicionarLivro()`, `atualizar()`) indicam métodos que devem ser implementados nas respectivas classes (Catálogo, Livro, Carrinho).
- **Diagrama de Sequência**: A direção das mensagens (por exemplo, Cliente → Catálogo para `selecionarLivro()`, Cliente → Carrinho para `adicionarLivro()`) pode ser visualizada em um diagrama de sequência, mostrando o fluxo de interações.
- **Refinamento do Diagrama de Classes**: Os métodos identificados refinam o diagrama de classes, especificando responsabilidades. Por exemplo:
  - Classe `Catálogo`: Método `selecionarLivro(livroId)`
  - Classe `Livro`: Método `exibirDetalhes()`
  - Classe `Carrinho`: Métodos `adicionarLivro(livro)` e `atualizar()`

4 Descrição do Diagrama de Sequência Exemplo

Com base na análise, um diagrama de sequência para o caso de uso pode ser descrito assim:

1. Cliente envia `selecionarLivro(livroId)` ao Catálogo.
2. Catálogo responde com detalhes do livro via `exibirDetalhes()` ao Cliente.
3. Cliente envia `adicionarLivro(livro)` ao Carrinho.
4. Carrinho processa a adição e chama `atualizar()` internamente.

5 Conclusão

Ao analisar o caso de uso "Comprar um Livro", extraímos objetos principais (Cliente, Livro, Catálogo, Carrinho), identificamos mensagens (`selecionarLivro()`, `exibirDetalhes()`, `adicionarLivro()`, `atualizar()`) e determinamos suas direções. Este processo conecta requisitos funcionais com o design orientado a objetos, possibilitando a criação de diagramas de sequência e classes que refletem o comportamento e a estrutura do sistema.

6 Próximos Passos

- Criar um diagrama de sequência para visualizar as interações.
- Refinar o diagrama de classes adicionando os métodos identificados e seus parâmetros.
- Validar o modelo contra casos de uso adicionais para garantir completude.